Amendment to Claims

1-13 (canceled).

14 (currently amended). A processor for executing instructions such that when the processor executes a first instruction and the processor determines after starting to execute the first instruction that the first instruction is blocked, the processor aborts execution of the first instruction and re-executes the first instruction. ~~accessing an unavailable resource, the processor suspends the first instruction and the processor circuitry which was to execute the first instruction becomes operable to execute one or more other instructions.~~

15 (currently amended). The processor Claim 32 [[14]] wherein the processor ~~executes~~ re-executes the first instruction ~~to completion~~ when the resource becomes available.

16 (canceled).

17 (currently amended). The processor of Claim 32 [[14]] wherein the processor performs multi-tasking, and the resource is unavailable for the first instruction if the resource is unavailable to a first task executing the first instruction. ~~becomes suspended when the first instruction is suspended, and while the task is suspended the processor circuitry that was to execute the first instruction is operable to execute one or more other tasks.~~

18 (currently amended). A multi-tasking processor comprising task scheduling circuitry, such that when a task TA1 executed by the processor ~~attempts~~ starts to execute a first instruction accessing a resource which is unavailable to the task TA1 due to the resource being made available to another task, ~~access an unavailable resource,~~ the task scheduling circuitry aborts execution of the first instruction after starting to execute the first instruction and suspends the task TA1 at least until the resource becomes available to the task TA1, and if another task TA2 is ready for execution in place of the task TA1 when the resource is unavailable to the task TA1, the task scheduling circuitry schedules the task TA2 ~~TA1~~,

wherein the task scheduling circuitry operation does not involve instruction execution by the processor.

19 (currently amended). ~~A multi-tasking processor comprising:~~

The processor of Claim 37 further comprising:

first circuitry for generating a first signal indicating whether [[a]] one or more task suspend conditions are ~~condition is~~ true for one or more tasks including the first task; and

second circuitry for scheduling a task or tasks for execution in response to the first signal.

20 (currently amended). The processor of Claim 19 further comprising third circuitry for generating a release signal indicating whether [[a]] one or more release conditions are ~~condition is~~ true for releasing [[a]] one or more tasks including the first task from the one or more task suspend ~~condition~~ conditions,

wherein the second circuitry is responsive to the release signal when the second circuitry schedules a task or tasks for execution.

21 (original). The processor of Claim 19 further comprising, for each task, a separate circuit for generating a signal SIG1 indicating whether the task is ready for execution, wherein the second circuitry is responsive to one or more signals SIG1 in scheduling a task or tasks for execution.

22 (original). The processor of Claim 19 wherein the second circuitry is to schedule a task or tasks for execution on each instruction executed by the processor such that whenever the processor is to execute any instruction, the second circuitry is to perform the task scheduling to schedule a task that will execute the instruction.

23 (currently amended). A method for executing computer instructions, the method comprising:

~~executing~~ starting to execute a first instruction; ~~accessing a computer resource;~~

after starting to execute the first instruction, receiving a signal indicating that the first instruction is blocked;

aborting execution of the first instruction in response to said signal and re-executing the first instruction.

~~if the resource is unavailable, then suspending the first instruction and executing one or more other instructions by circuitry which was to execute the first instruction.~~

24 (currently amended). The method of Claim 52 ~~23 further comprising executing~~ wherein the first instruction is re-executed ~~to completion~~ when the resource becomes available.

25 (currently amended). The method of Claim 52 [[23]] wherein executing the first instruction comprises executing the first instruction by a first task, and the resource is unavailable to the first task.

~~when the first instruction is suspended, the first task is suspended and execution of the one or more other instructions comprises execution of one or more other tasks.~~

26 (currently amended). ~~A multi-tasking method comprising:~~

The method of Claim 57 wherein ~~generating~~ said signal comprises a first signal indicating whether [[a]] one or more task suspend conditions are ~~condition is~~ true for one or more tasks including the first task;

wherein the method further comprises scheduling a task or tasks for execution in response to the first signal.

27 (currently amended). The method of Claim 26 further comprising generating a release signal indicating whether [[a]] one or more release conditions are ~~condition is~~ true for releasing [[a]] one or more tasks including the first task from the one or more task suspend ~~condition~~ conditions,

wherein scheduling a task or tasks for execution is responsive to the release signal.

28 (original). The method of Claim 26 further comprising generating, for each task, a separate signal indicating whether the task is ready for execution.

29 (original). The method of Claim 26 wherein scheduling a task or tasks for execution is performed on each instruction executed by any one of the tasks such that whenever an instruction is to be executed, the task scheduling is performed to schedule a task that will execute the instruction.

30 (new). The processor of Claim 14 wherein:

the processor comprises an instruction execution unit for executing instructions stored in a memory and fetched from the memory;

the processor aborts the first instruction after the first instruction has been fetched from the memory; and

the processor again fetches the first instruction from the memory to re-execute the first instruction.

31 (new). The processor of Claim 14 wherein the first instruction is aborted after being decoded by the processor, and the first instruction is decoded again when the first instruction is being re-executed by the processor.

32 (new). The processor of Claim 14 wherein the first instruction is blocked if the first instruction is to access a resource unavailable for the first instruction.

33 (new). The processor of Claim 32 wherein the resource is a storage area.

34 (new). The processor of Claim 33 wherein the storage area comprises a request FIFO for storing requests to process network data, and the first instruction is an instruction to read the request FIFO.

35 (new). The processor of Claim 33 wherein the storage area comprises a command FIFO for storing commands for processing of network data, and the first instruction is an instruction to write one or more commands to the command FIFO.

36 (new). The processor of Claim 33 wherein the storage area comprises a status FIFO for storing status information on reception of network data over a network, and the first instruction is an instruction to read the status FIFO.

37 (new). The processor of Claim 17 wherein when the first instruction is aborted, the processor is operable to suspend the first task and to execute another task instead of the first task while the first task is suspended.

38 (new). The processor of Claim 37 wherein suspension of the first task and scheduling of the other task instead of the first task does not involve instruction execution by the processor.

39 (new). The processor of Claim 37 wherein the first task remains suspended at least until the resource becomes available to the first task.

40 (new). The processor of Claim 17 wherein the processor is operable to execute in parallel a plurality of groups of tasks, wherein each group of tasks is associated with one or more network flows to process data in the one or more network flows, wherein the first task belongs to a first group of tasks which is one of the groups of tasks, and when the first instruction is aborted, the processor is operable to suspend the first task and to execute another task in the first group of tasks instead of the first task.

41 (new). The processor of Claim 17 wherein when the first instruction is aborted, the processor is operable not to suspend the first task and to re-execute the first instruction for the first task without suspending the first task.

42 (new). The processor of Claim 17 wherein the processor comprises an instruction execution pipeline operable to concurrently execute instructions from different tasks, wherein the processor is operable to abort the first instruction after the pipeline starts execution of one or more other instructions for one or more tasks other than the first task but before the pipeline starts execution of any instruction following the first instruction for the first task.

43 (new). The processor of Claim 42 wherein the pipeline comprises an instruction decode stage, and the first instruction is aborted after being processed by the decode stage.

44 (new). The processor of Claim 42 wherein the first instruction is aborted in the pipeline's read stage in which instruction operands are read from storage and presented to instruction execution logic.

45 (new). The processor of Claim 17 wherein the resource is shared by the first task and at least one other task.

46 (new). The processor of Claim 45 wherein the resource is a storage area.

47 (new). The processor of Claim 46 wherein the storage area comprises a request FIFO for storing requests to process network data, and the first instruction is an instruction to read the request FIFO.

48 (new). The processor of Claim 46 wherein the storage area comprises a command FIFO for storing commands for processing of network data, and the first instruction is an instruction to write one or more commands to the command FIFO.

49 (new). The processor of Claim 46 wherein the storage area comprises a status FIFO for storing status information on reception of network data over a network, and the first instruction is an instruction to read the status FIFO.

50 (new). The method of Claim 23 wherein said signal is received after the first instruction has been fetched from a memory for execution; and

the method further comprises fetching the first instruction again from the memory to re-execute the first instruction.

51 (new). The method of Claim 23 wherein the first instruction is aborted after being decoded, and the first instruction is decoded again when the first instruction is being re-executed.

52 (new). The method of Claim 23 wherein the first instruction is blocked due to the first instruction attempting to access a resource unavailable for the first instruction.

53 (new). The method of Claim 52 wherein the resource is a storage area.

54 (new). The method of Claim 53 wherein the storage area comprises a request FIFO for storing requests to process network data, and the first instruction is an instruction to read the request FIFO.

55 (new). The method of Claim 53 wherein the storage area comprises a command FIFO for storing commands for processing of network data, and the first instruction is an instruction to write one or more commands to the command FIFO.

56 (new). The method of Claim 53 wherein the storage area comprises a status FIFO for storing status information on reception of network data over a network, and the first instruction is an instruction to read the status FIFO.

57 (new). The method of Claim 25 further comprising, when the first instruction is aborted, suspending the first task and executing one or more other tasks instead of the first task while the first task is suspended.

58 (new). The method of Claim 57 wherein suspension of the first task and scheduling of each of the one or more other tasks instead of the first task does not involve instruction execution.

59 (new). The method of Claim 57 wherein the first task remains suspended at least until the resource becomes available to the first task.

60 (new). The method of Claim 25 wherein when the first instruction is aborted, the first task is not suspended and the first instruction is re-executed without suspending the first task.

61 (new). The method of Claim 25 wherein the first instruction is aborted while being executed in an instruction execution pipeline after the pipeline starts execution of one

or more other instructions for one or more tasks other than the first task but before the pipeline starts execution of any instruction following the first instruction for the first task.

62 (new). The method of Claim 61 wherein the pipeline comprises an instruction decode stage, and the first instruction is aborted after being processed by the decode stage.

63 (new). The method of Claim 61 wherein the first instruction is aborted in the pipeline's read stage in which instruction operands are read from storage and presented to instruction execution logic.

64 (new). The method of Claim 25 wherein the resource is shared by the first task and at least one other task.

65 (new). The method of Claim 25 wherein the resource is a storage area.

66 (new). The method of Claim 65 wherein the storage area comprises a request FIFO for storing requests to process network data, and the first instruction is an instruction to read the request FIFO.

67 (new). The method of Claim 65 wherein the storage area comprises a command FIFO for storing commands for processing of network data, and the first instruction is an instruction to write one or more commands to the command FIFO.

68 (new). The method of Claim 65 wherein the storage area comprises a status FIFO for storing status information on reception of network data over a network, and the first instruction is an instruction to read the status FIFO.